

DTIC FILE COPY

RADC-TR-89-294
In-House Report
October 1989



(4)

AD-A217 542

OPTIMAL PATH ANALYZER (VERSION 1.0)

Scott M. Huse

DTIC
ELECTE
FEB 02 1990
S D D

"Original contains color.
plates: All DTIC reproductions
will be in black and
white."

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

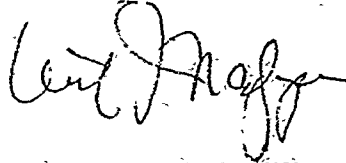
ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, NY 13441-5700

9 0 02 02 0 52

This report has been reviewed by the RADC Public Affairs Office (PA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

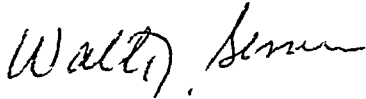
RADC TR-89-294 has been reviewed and is approved for publication.

APPROVED:



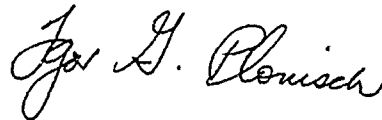
VINCENT J. NARDOZZA
Chief, Image Systems Division
Directorate of Intelligence and Reconnaissance

APPROVED:



WALTER J. SENUS
Technical Director
Directorate of Intelligence and Reconnaissance

FOR THE COMMANDER:



IGOR G. PLONISCH
Directorate of Plans and Programs

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (IRRA) Griffiss AFB NY 13441-5700. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document requires that it be returned.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b. RESTRICTIVE MARKINGS N/A		
2a. SECURITY CLASSIFICATION AUTHORITY N/A			3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.		
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A					
4. PERFORMING ORGANIZATION REPORT NUMBER(S) RADDC-TR-89-294			5. MONITORING ORGANIZATION REPORT NUMBER(S) N/A		
6a. NAME OF PERFORMING ORGANIZATION Rome Air Development Center		6b. OFFICE SYMBOL (if applicable) IRRA	7a. NAME OF MONITORING ORGANIZATION Rome Air Development Center (IRRA)		
6c. ADDRESS (City, State, and ZIP Code) Griffiss AFB NY 13441-5700			7b. ADDRESS (City, State, and ZIP Code) Griffiss AFB NY 13441-5700		
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Rome Air Development Center		8b. OFFICE SYMBOL (if applicable) IRRA	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N/A		
8c. ADDRESS (City, State, and ZIP Code) Griffiss AFB NY 13441-5700			10. SOURCE OF FUNDING NUMBERS		
		PROGRAM ELEMENT NO 62702F	PROJECT NO. 4594	TASK NO 18	WORK UNIT ACCESSION NO. N3
11. TITLE (Include Security Classification) OPTIMAL PATH ANALYZER (VERSION 1.0)					
12. PERSONAL AUTHOR(S) Scott M. Huse					
13a. TYPE OF REPORT In-House		13b. TIME COVERED FROM Jun 89 TO Jul 89	14. DATE OF REPORT (Year, Month, Day) October 1989		15. PAGE COUNT 42
16. SUPPLEMENTARY NOTATION N/A					
17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP			
12	01				
15	04				
			Artificial neural network Perceptron		
			Optimal path Neuron		
19. ABSTRACT (Continue on reverse if necessary and identify by block number)					
<p>This report describes the function, operation, test and evaluation results of the Optimal Path Analyzer (Version 1.0), a locally connected artificial neural network that maps an optimal path from a given starting point to a given destination through a field of obstacles.</p>					
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT <input type="checkbox"/> DTIC USERS			21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a. NAME OF RESPONSIBLE INDIVIDUAL SCOTT HUSE			22b. TELEPHONE (Include Area Code) (315) 330-3176		22c. OFFICE SYMBOL RADDC (TRRA)

DD Form 1473, JUN 86

Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

TABLE OF CONTENTS

<u>Section</u>	<u>Page</u>
Introduction.....	1
Background.....	13
Scope.....	14
Technical Description.....	14
Test and Evaluation.....	16
Recommendations.....	20
References.....	30

Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution	
Availability Codes	
Dist	Avail and for Special
A-1	



LIST OF FIGURES

<u>Figure</u>	<u>Page</u>
1. Optimal Path Analyzer Map.....	2
2. Interconnection Weight Assignment Table.....	3
3. An Artificial Neuron.....	9
4. A Basic Neural Network.....	10
5. Output Iteration Test Results.....	19
6. OPA 1.0 Sample Run B3 - S3.....	21
7. OPA 1.0 Sample Run B1 - T6.....	22
8. OPA 1.0 Sample Run B9 - P1.....	23
9. OPA 1.0 Sample Run C6 - S8.....	24
10. OPA 1.0 Sample Run N1 - C3.....	25
11. OPA 1.0 Sample Run N6 - T9.....	26
12. OPA 1.0 Sample Run A0 - P0.....	27
13. OPA 1.0 Sample Run I0 - T6.....	28
14. OPA 1.0 Sample Run C7 - J7.....	29

OPTIMAL PATH ANALYZER (VERSION 1.0)

I. Introduction

A. Objective

The purpose of this project was to design a locally connected artificial neural network that would map an optimal/near optimal path from any given starting point to any given destination through a field of obstacles. This was accomplished by the program, "Optimal Path Analyzer" (version 1.0), or OPA 1.0.

B. Approach

A simplified map was drawn (Figure 1) and superimposed on a grid field representing a locally connected neural network. Interconnection weights were fixed values based upon the ease or difficulty of movement from one neuron to another. Figure 2 summarizes these values.

Initially, the output values of all processing elements were set to 1 with two noteworthy exceptions: (1) those located with the mountain, lake, quicksand pit, and RADC building were set to 0 and, (2) the destination was set to 1000.

(K R) ←

Optimal Path Analyzer

A B C D E F G H I J K L M N O P Q R S T

0

1

2

3

4

5

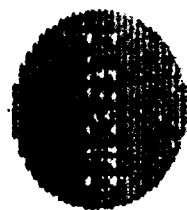
6

7

8

9

00000000



Origin coordinates (A-T 0-9) :

Figure 1. Optimal Path Analyzer map.

OPTIMAL PATH ANALYZER

Interconnection Weights

Feature	Value
Primary road	.90
Primary road on hill	.85
Dirt road	.80
Open/Flat countryside	.60
Countryside hill	.40
Shallow stream	.30
Deep stream	.20
Mountain	.00
RADC building	.00
Lake	.00
Quicksand	.00

Figure 2. Interconnection weight assignment values.

The destination neuron's output value was iteratively propagated throughout the network in a sequential manner. This was done row by row starting with the processing element in row 1, column 1 (upper-left corner) progressing down to the processing element in row 10, column 20 (lower-right corner).

Once the field was propagated throughout the entire network, each neuron was color-coded according to its respective output level. Then, an optimal path line was drawn from the starting point to the destination. The path was locally determined at each step by comparing the output values of the northern, southern, eastern and western neighboring neurons. A line segment was iteratively drawn to the neuron whose output value was highest until the destination point was reached.

The following pages of this technical report include:

- A brief history of neural networks
- A short generalized course on neural networks
- Project background, scope, technical description, testing, and evaluation results
- Recommendations
- Sample runs
- References

C. Neural Networks - A Brief History

The modern revitalized interest in neural network research and development is relatively new. However, work actually began more than 40 years ago.

McCulloch and Pitts (1943) demonstrated how neural-like networks could compute. Then, in 1949, Donald Hebb proposed a concept that profoundly influenced the field. His idea is now known as Hebb's Law and essentially states that if a neuron, say X, is repeatedly stimulated by another neuron, Y, then neuron X will become more sensitive to stimuli from neuron Y. Consequently, the synaptic connection from Y to X will be more efficient and it will now be easier for Y to stimulate X to produce an output in the future.

In 1951, Dean Edmonds and Marvin Minsky built a learning machine. In 1958, Rosenblatt invented a class of simple neuron-like networks which he called "perceptrons".

In 1969 algorithmic programming took the lead in the field of AI. This shift is commonly attributed to Minsky and Papert's book, Perceptrons (1969), which discouraged further neural network research because restricted types of networks were proven unable to solve certain types of problems (e.g., computing the exclusive-or function).

Nevertheless, during the late '70s, several researchers continued to study the neuron concept. Then, finally, in the early '80s, John Hopfield presented a paper at the National Academy of Sciences which essentially resurrected the concept of neural networks back from the dead.

Recently, interest in neural networks has grown exponentially making up for the 20-year Rip Van Winkle snooze. Numerous companies, including giants like IBM, AT&T, Texas Instruments and Fujitsu, are conducting extensive research and development in this field.

The Defense Advanced Research Projects Agency (DARPA) recently released a study that recommended neural network funding of approximately \$400 million over an eight year period. The study's long-range goal is to achieve an artificial neural network with 10^{10} connections that can run at a rate of 10^{12} ips. In comparison, today's computer-based neural simulations can only run at about 10^7 ips.

Why all the stir? One reason is that, while traditional computing and conventional AI approaches are successful in many areas, they fail at certain tasks which a mere child can perform with ease, e.g., pattern recognition, text-to-speech conversion, natural language processing, and so forth. Such

tasks are often combinatorially explosive and, if handled by traditional methods, typically require an extraordinary number of rules and instructions.

Neural networks, however, do not require such programming; they can be trained rather than programmed in the classical sense. Like living systems, they process information in a dynamic, self-organizing way and they can learn from experience. Also, due to massive parallelism, neural networks can make high-speed decisions and are potentially fault-tolerant.

Neural networks can be implemented in hardware and/or software. There are at least fifty different types of networks that are either being explored in research or being developed for applications. Although Bell Labs and a few others are concentrating on neural network chips, the most common research and development approach is a software simulation of a neural network that is run on a conventional processor.

An interesting and truly unusual aspect of the neural network field of study is its interdisciplinary nature. Interest is shared by engineers, computer scientists, neurophysiologists, psychologists, optical specialists,

philosophers, mathematicians, and others.

The field of neural networks is a potent emerging technology. It is finding more and more practical applications and the potential for this area looks promising.

D. Neural Networks - A Short Generalized Course

A neural network may be defined as a complex system composed of many simple processing elements (neurons) operating in parallel whose function is determined by network structure, connection strengths, and the processing performed at the neurons.

The basic artificial neuron (Figure 3) typically receives multiple inputs, processes those inputs, and then emits an output signal which is distributed throughout the network. When grouped together, these simple elements form layers. An example of a three-layer artificial neural network is given in Figure 4. Taken collectively, these layers form the overall structure of a neural network.

The three essential components of an artificial neural network are the architecture, the transfer function, and the learning rule. The architecture defines the basic flow of

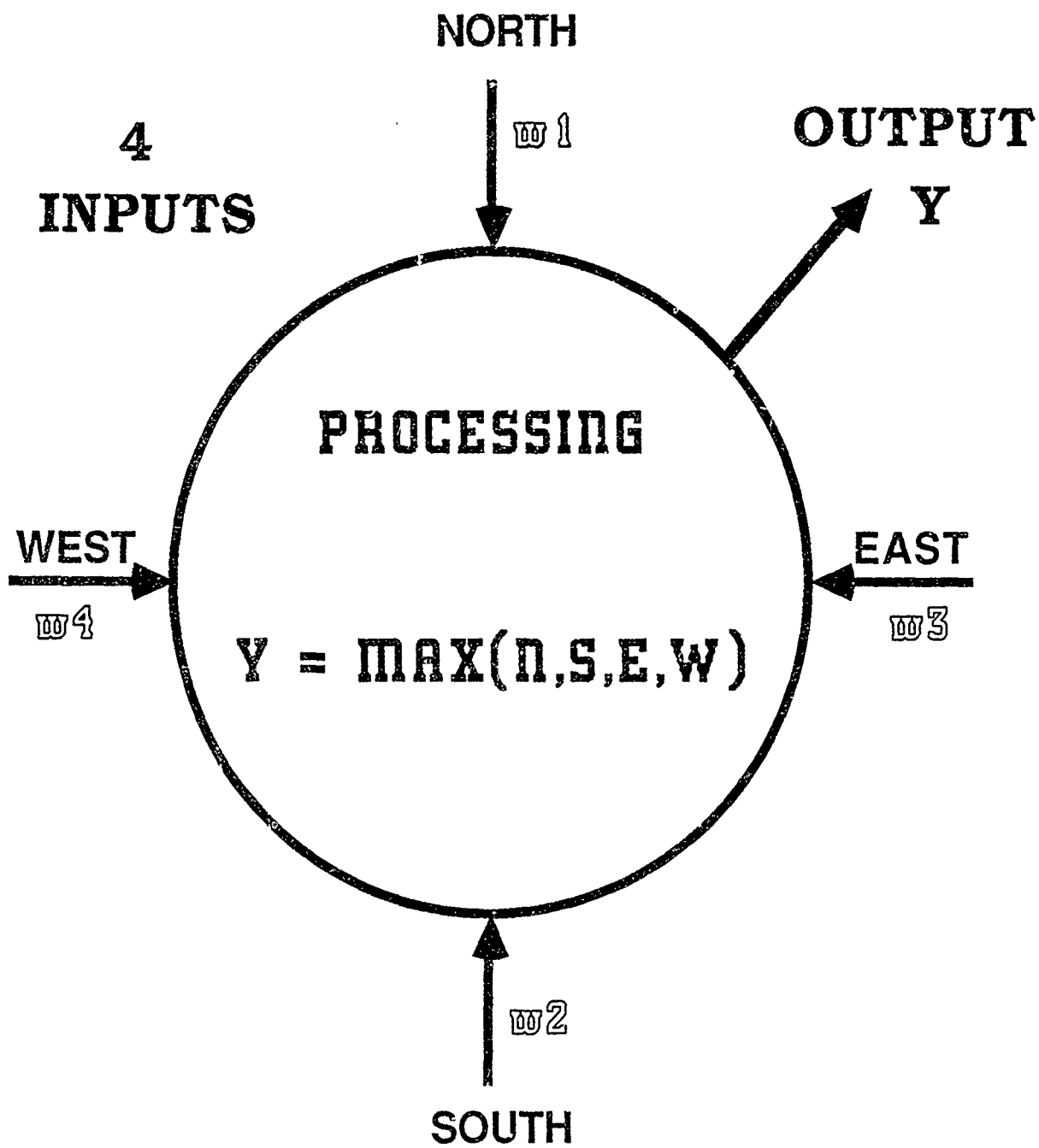


FIGURE 3. A single artificial neuron.

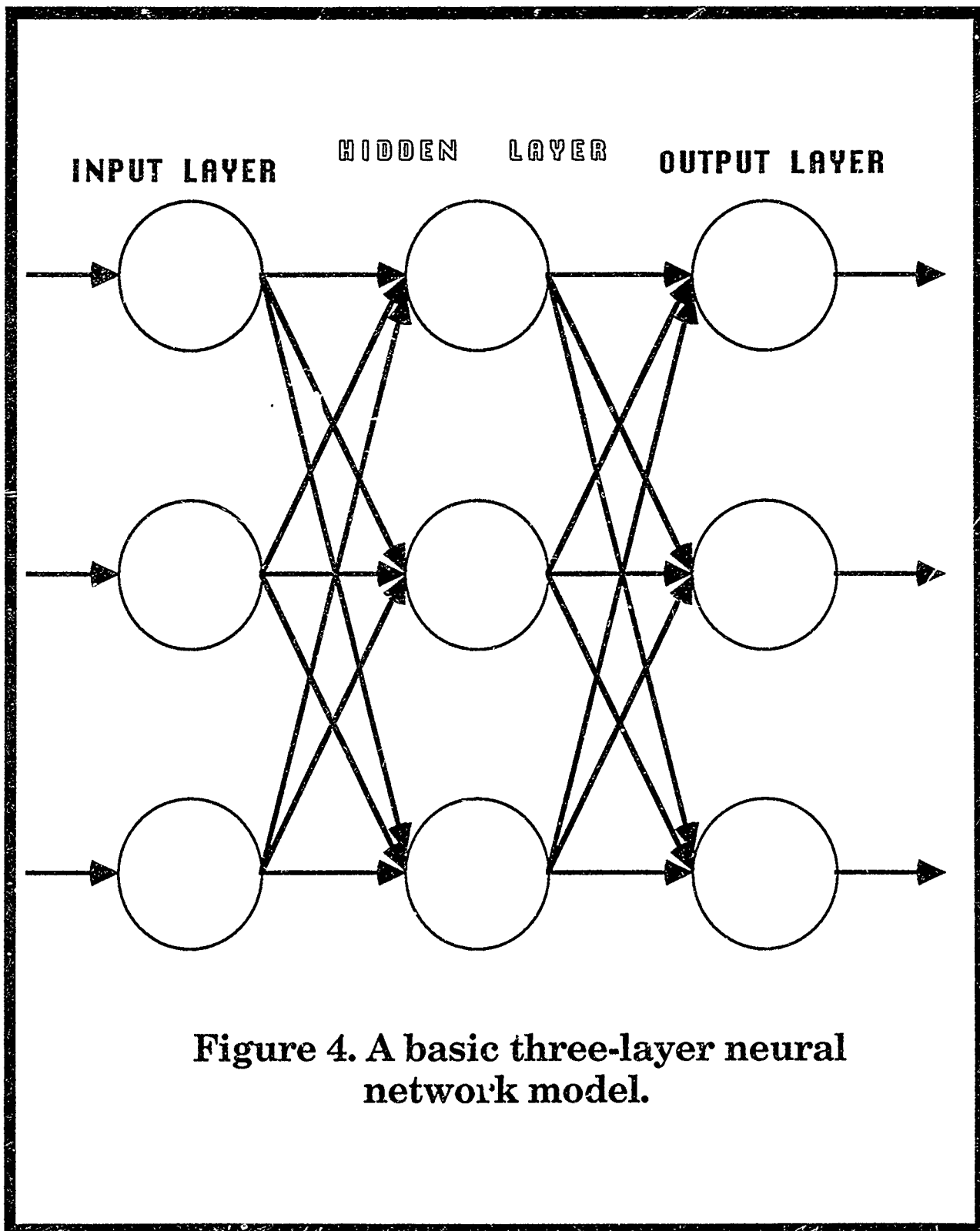


Figure 4. A basic three-layer neural network model.

information in the network. The transfer function determines how input information and interconnection weights are used to calculate an output value. The learning rule specifies how interconnection weights are to be adjusted in the process of training the network.

The kinds of problems a given network can solve depends on which network paradigm is used. For example, back-propagation is good for approximating functional relationships and pattern-classifying expert systems. Adaptive resonance-theory networks are good at classifying a wide range of patterns with widely varying information content.

Networks that allow changes in the synaptic weights are called "learning networks". Some networks, such as OPA 1.0, use fixed precalculated interconnection weights and do not apply a learning rule.

There are two types of learning: supervised and unsupervised. Supervised learning procedures require some sort of teacher or expert to specify what the desired output should be. Unsupervised learning procedures try to develop internal models to capture regularities in the input signals. Supervised learning procedures are presently more popular simply because they have attained good results.

A neural network has to be trained before it can become operational. This training consists of presenting a set of inputs and, for each input, presenting the corresponding output that is expected from the network. During the training phase, internal weights are adjusted to produce the required output. The training phase is considered complete when the network produces the required outputs (within some error criterion) for the training set. There are numerous learning rules including Hebb's rule, the Delta rule, the Back-propagation procedure, Kohonen's self-organization learning procedure, Drive-reinforcement, and so forth.

One important fact to remember when working with artificial neural networks is that the real knowledge of the network is generally stored in the interconnection weights of the network. It is not stored in the outputs, contrary to what one might suppose intuitively.

A massive neural network - one containing at least thousands of neurons with millions of interconnections and having feedback capabilities is needed to realistically simulate neural networks found in nature. Consider, for example, that a worm's brain is estimated to have 1000 interconnected neurons; a human's around 100 billion.

It has been said that it takes over half your brain to be able to walk and chew gum at the same time. How many neurons does it take for you to change a light bulb? That's right: "billions and billions."

II. Background

The Battlefield Intelligence Branch (IRRA) of the Intelligence & Reconnaissance Directorate at RADC is interested in the design and implementation of an artificial neural network that will perform a multiple constraint satisfaction in polynomial time, for the purpose of doing a trafficability analysis and route prediction for a fixed and/or mobile target.

Optimal Path Analyzer 1.0 (OPA 1.0) is a first-step prototype designed to demonstrate proof of concept with a neural network model. OPA 2.0 is currently being developed and should be implemented soon on a SUN workstation. Present plans call for OPA 2.0 to take digital map and feature data as input and plot optimal/near optimal paths based a variety of terrain, feature, and doctrinal constraints.

III. Scope

This report presents an overview of the Optimal Path Analyzer (version 1.0) and includes a brief technical description, test results, evaluations and recommendations. Areas of practical application that could be supported by this paradigm include mission planning, route prediction, and trafficability analysis.

IV. Technical Description

OPA 1.0 is a fixed-weights, nearest-neighbor neural network paradigm designed and programmed by Scott M. Huse. It was written in Turbo C 2.0 on a 286 Wyse PC. The executable file (OPA.EXE), accompanied by the file TEST.DTA, should run on any PC-compatible machine equipped with a graphics adapter and EGA monitor.

A simplified map was drawn and superimposed on a grid field of 264 grid points which correspond to the processing elements, or neurons, of the network. Each neuron was locally connected to its respective neighbors to the north, south, east and west. Interconnection weights were fixed values whose magnitudes corresponded to the ease or difficulty

of movement from one neuron to another (Figure 2).

The output values of each neuron were initially set to 1, with two noteworthy exceptions: (1) Neurons located within the lake, mountain, quicksand pit and RADC building which were set to 0 and, (2) the destination neuron's output was set to 1000.

Once the user has entered legitimate starting and destination coordinates, the destination neuron's output value is systematically propagated row by row throughout the entire network starting from the upper left-most neuron of the map down to the lower right-most neuron. The output value for each neuron is determined by selecting the maximum product value of its neighboring neuron's outputs and interconnection weights. This process is iterated 29 times in order to insure that each neuron receives direct or indirect input from the destination node's output, at least once.

When this process is completed, each neuron is color-coded according to its output level. Neurons with outputs ranging from 0 to 10 are colored light red (unfavorable area); neurons with outputs 11 - 100 are colored yellow (favorable area); and those whose outputs range from 101 - 1000 are colored light blue (most favorable area).

It is clear from the sample runs (Figures 6-14) that the coloring scheme distributions are entirely dependent upon the relative locations of the starting and destination points. Further, it is apparent how "output shadows" are cast behind obstacles as the network is propagated. This color coding scheme helps the user to visualize how the destination output was propagated throughout the network and why the network selected the particular path that it did.

Finally, a color-coded optimal/near optimal path line is drawn step-by-step from the starting point to the destination. The decision as to where to draw the line is determined locally by selecting the neuron to the north, south, east, or west which is broadcasting the highest output value. This process is repeated until the destination point is reached.

V. Test and Evaluation

An important question that had to be answered with regard to this particular neural network was, "What was the minimum number of iterations needed to propagate the destination neuron's output so that every neuron in the network would receive input (directly or indirectly) from the destination

neuron's output, and thereby ensure that an optimal/near optimal path would be achieved?"

First of all, consider that although the network consists of a 22 by 12 neuron rectangular grid, the border neurons are not within the map's boundaries. Therefore, practically speaking, the actual data map consists of a 20 by 10 neuron rectangular grid.

The longest possible path between any two given points on this particular map (Figure 1) is from one corner to its opposite corner, travelling along two sides of the perimeter of the map. Consequently, a worst-case scenario, in terms of output propagation distance, would require the traversing of 30 neurons (29 interconnections). Therefore, the destination neuron's output must be propagated throughout the network 29 times in order to guarantee that every neuron in the network will receive input from the destination neuron's output, and thereby ensure that an optimal/near optimal path will be drawn for every possible legitimate coordinate combination.

In order to test this hypothesis and better understand the effects of varying the number of iterations, tests were run in which the number of destination output propagation iterations were varied using values of 1, 5, 10, 15, 20, 25

and 30. For each of these iteration tests, 100 starting and destination locations were randomly generated. The resultant paths were evaluated as either a success or failure. A path was classified as a success if an evidently optimal/near optimal path was achieved, and a failure otherwise. The results of these tests are summarized in Figure 5.

Although the results of this particular experiment suggest that 20 iterations are sufficient to obtain a 100% success rate, we know from the preceding calculations that a minimum of 29 iterations is required to absolutely guarantee an optimal/near optimal path for this particular map configuration.

Further, more direct testing revealed that failures do in fact occur with iterations as high as 27 with this particular map configuration. In general, the formula for calculating the appropriate number of destination output iterations for this type of network equals the sum of the number of rows and columns, minus 1 ($10 + 20 - 1 = 29$ for OPA 1.0).

When running this artificial neural network program, it is readily apparent that OPA 1.0 is quite capable of consistently mapping paths which are evidently optimal/near

OPTIMAL PATH ANALYZER

Output Iteration Test Results

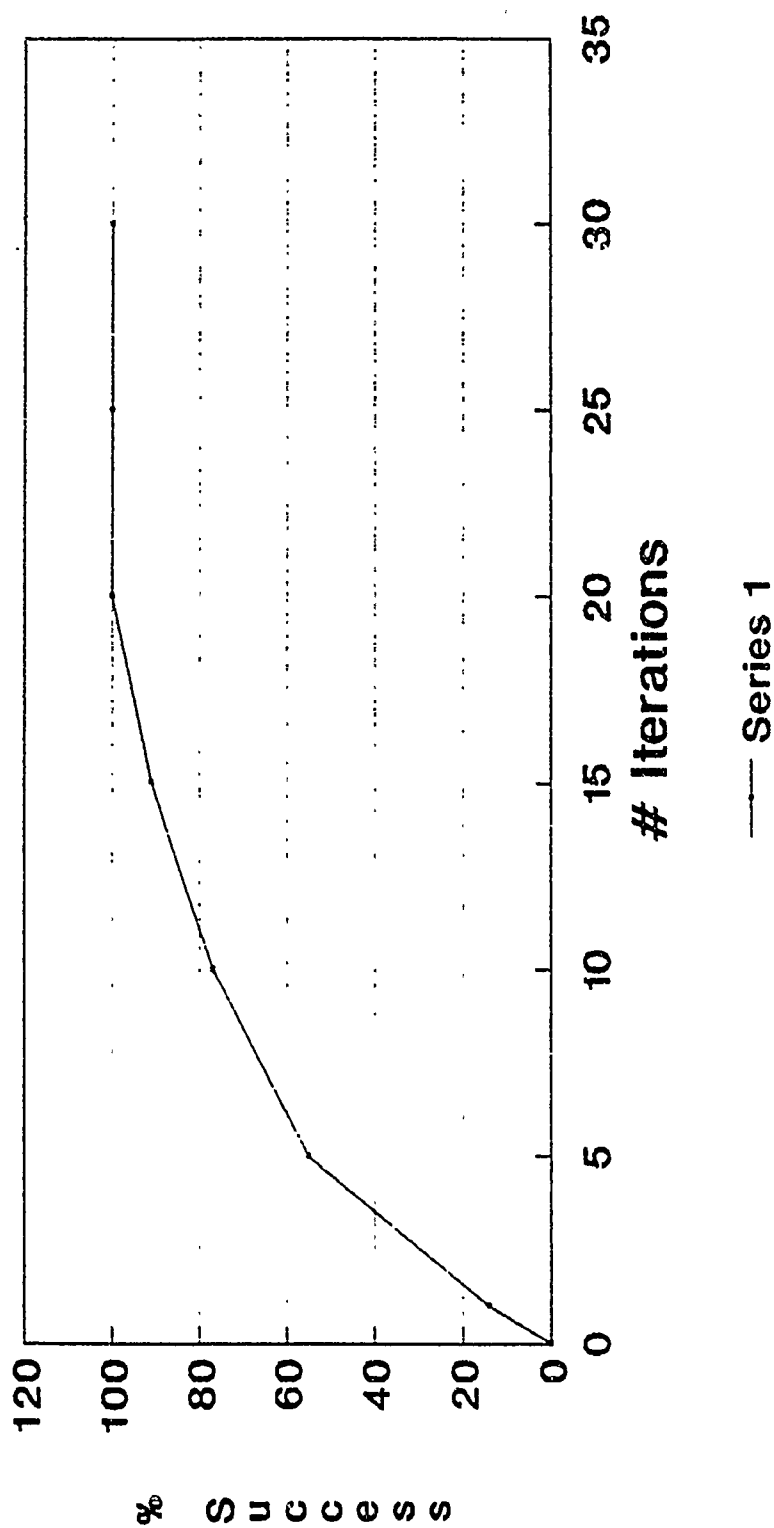


Figure 5. Output iteration test results.

optimal regardless of the given coordinates (Figures 6-14).

It is also instructional to note that OPA 1.0 generally prefers to follow roads rather than open/flat countryside. This "knowledge" is inherent in the interconnection weights. If the synaptic connections of the roads were weighted less favorably, OPA 1.0 would map more direct paths along the countryside. This confirms the fact that network behavior is determined by the interconnection weight knowledge, not the outputs of the neurons.

VI. Recommendations

It is recommended that OPA 1.0 be enhanced and expanded into OPA 2.0 which would be a more versatile and practical SUN workstation version. Present plans call for OPA 2.0 to take actual digital map and feature data as input and plot optimal/near optimal paths based upon a variety of terrain, feature and doctrinal constraints.

Optimal Path Map

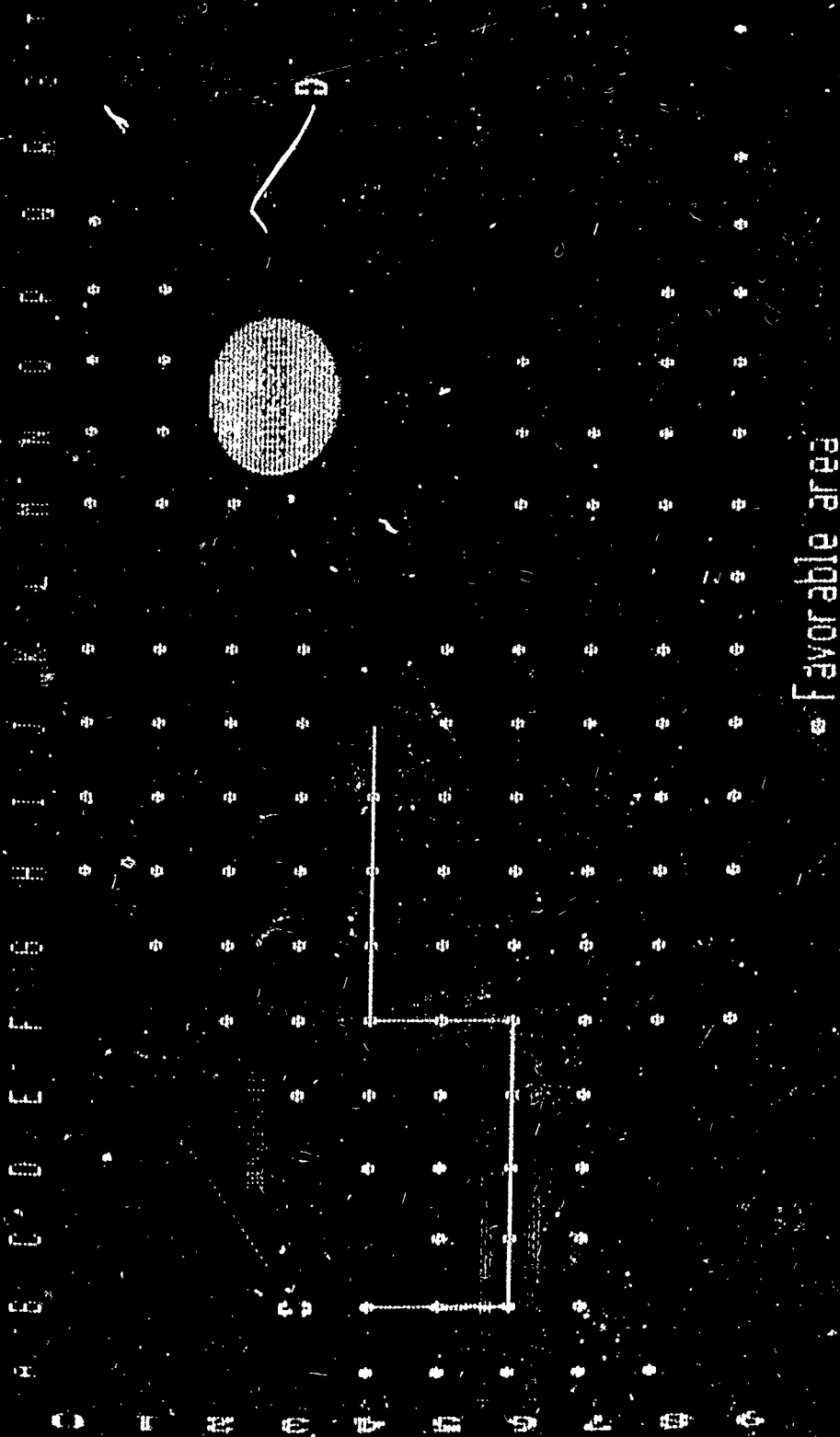


Figure 6. OPA 1.0 sample run B3 - S3.

Optimal Path Analyzer

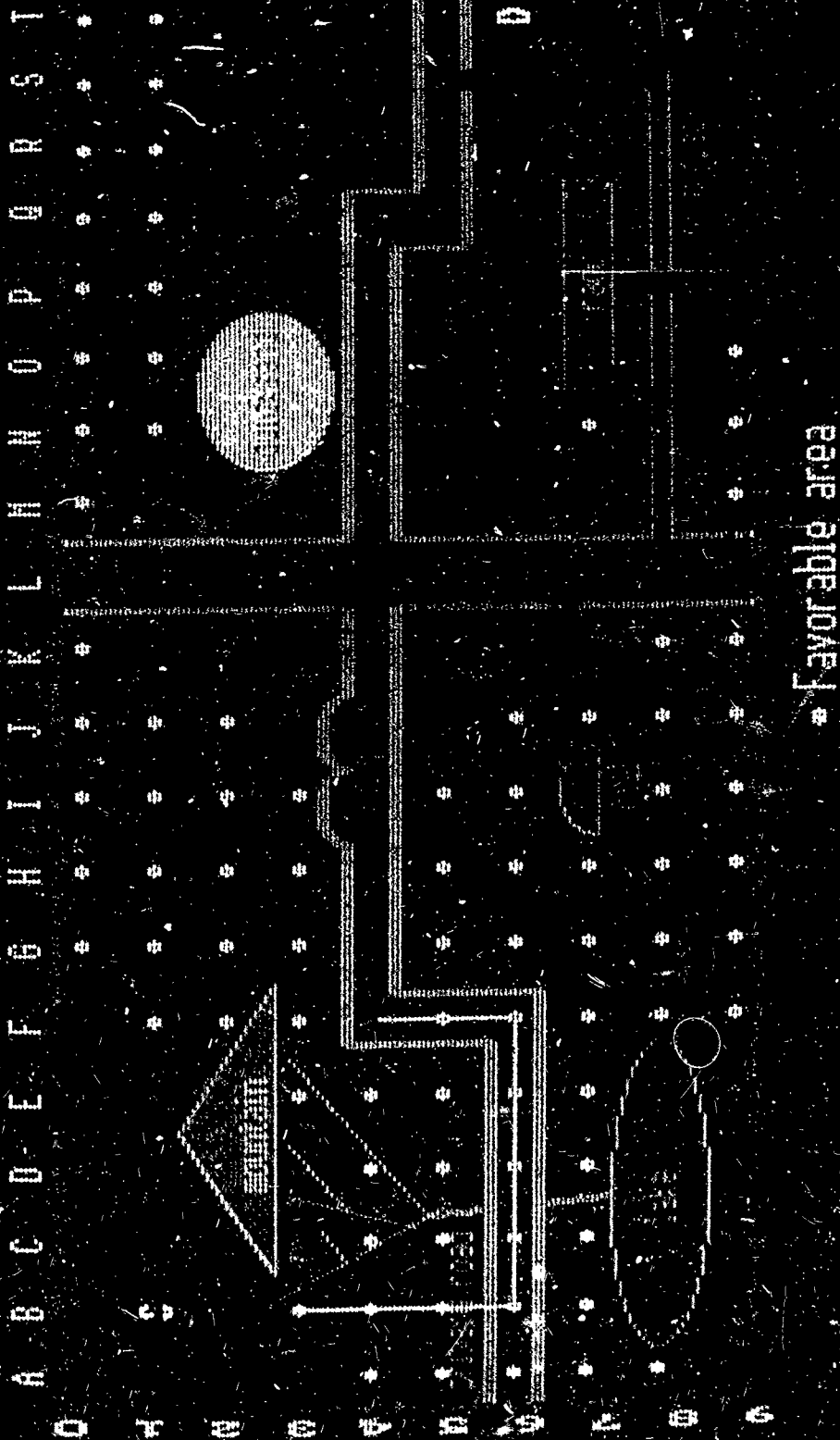


Figure 7. OPA 1.0 sample run B1 - T6.

Optimal Path Analyzer

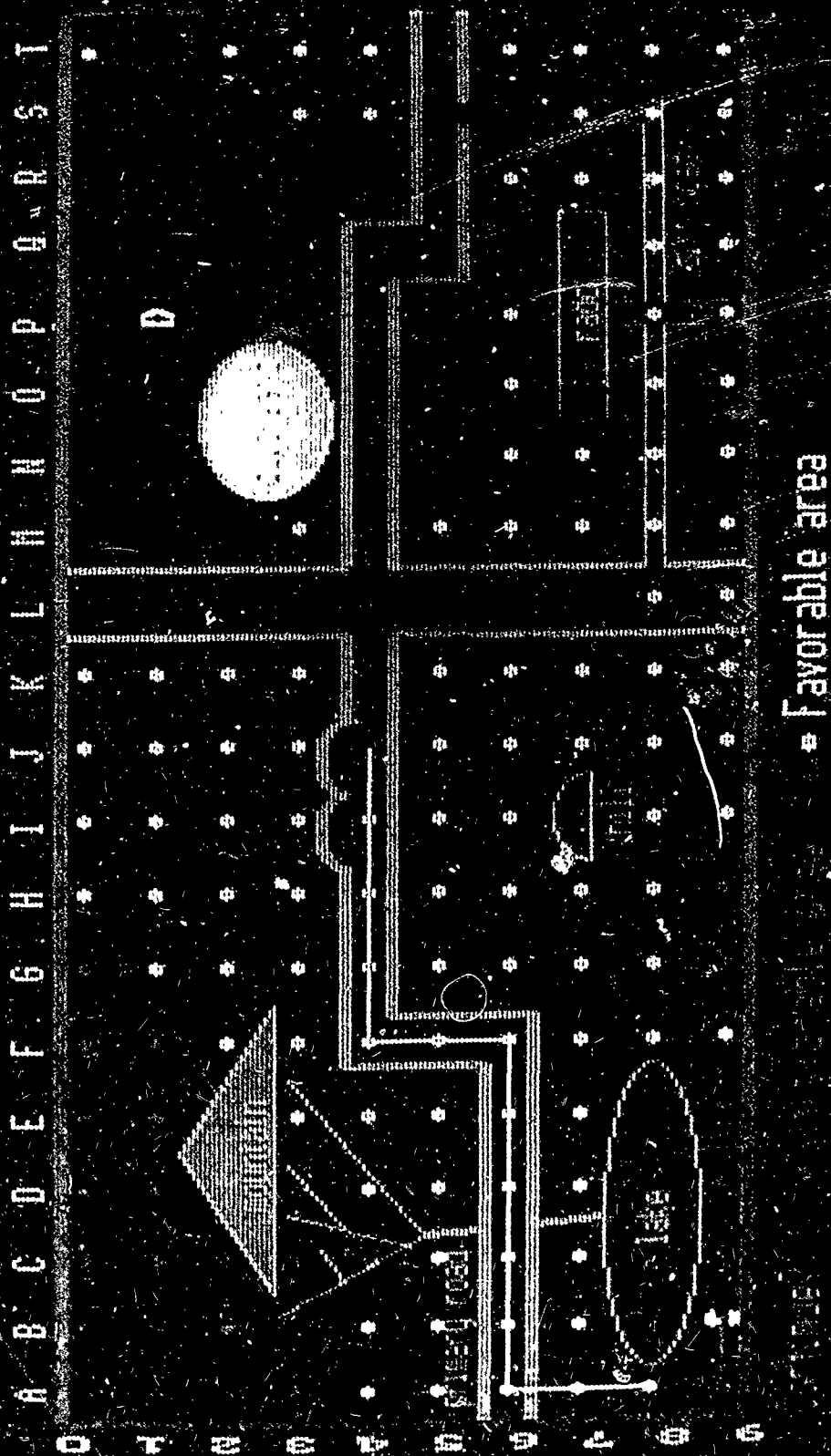
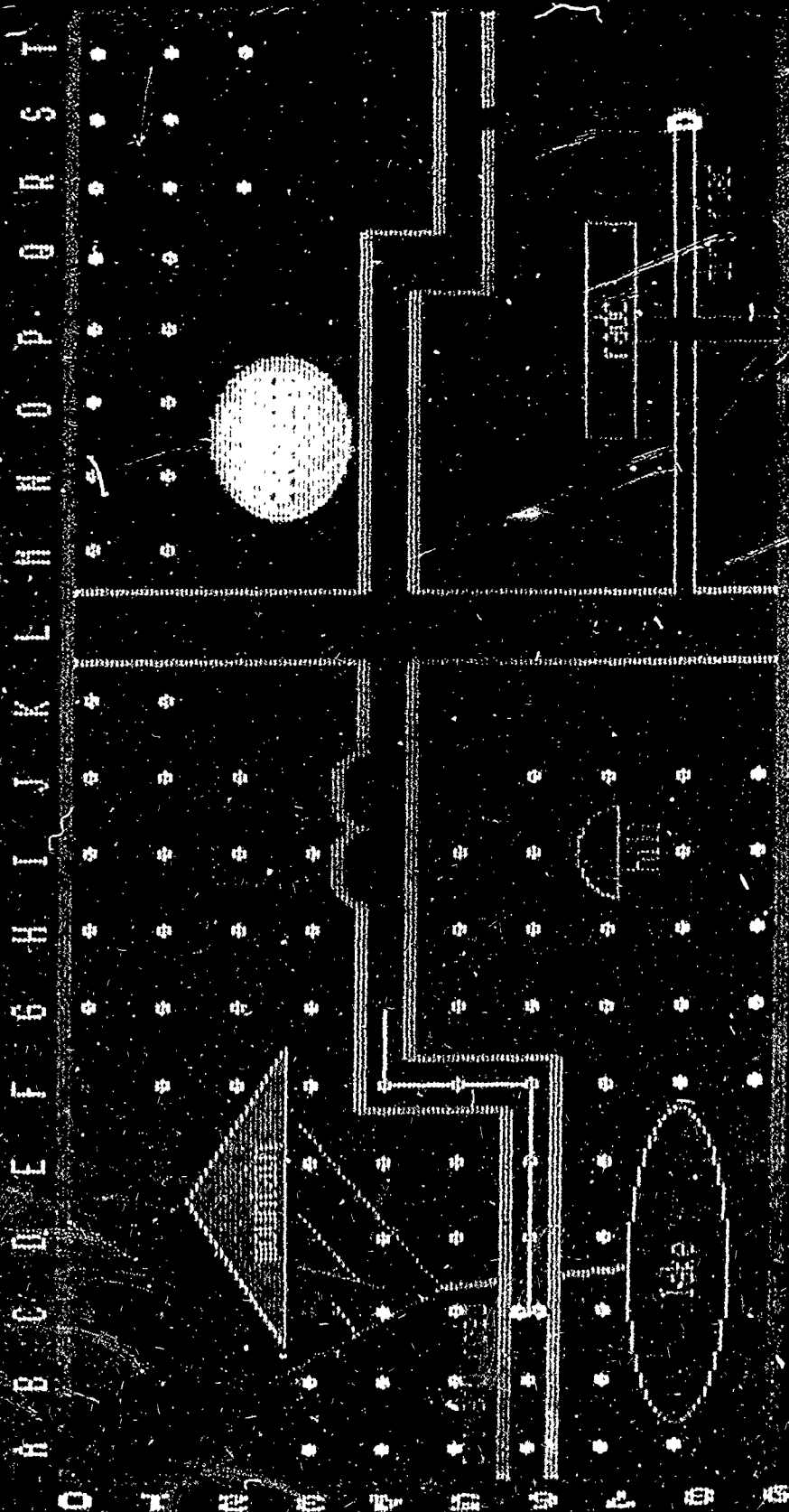


Figure 8. OPA 1.0 sample run B9 - P1.

Optimal Path Analyzer



• Favorable area

Figure 9. OPA 1.0 sample run C6 - S8.

Optimal Path Analyzer

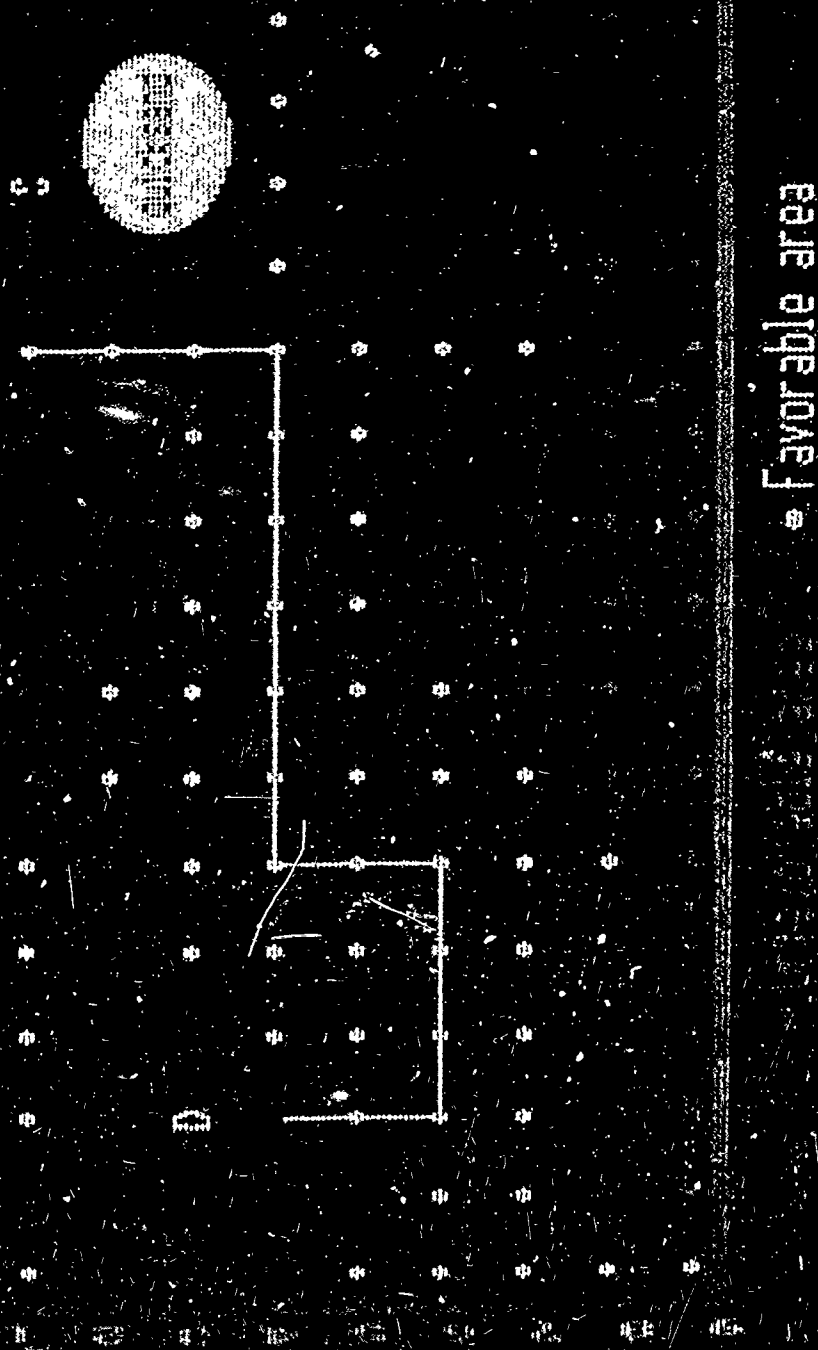
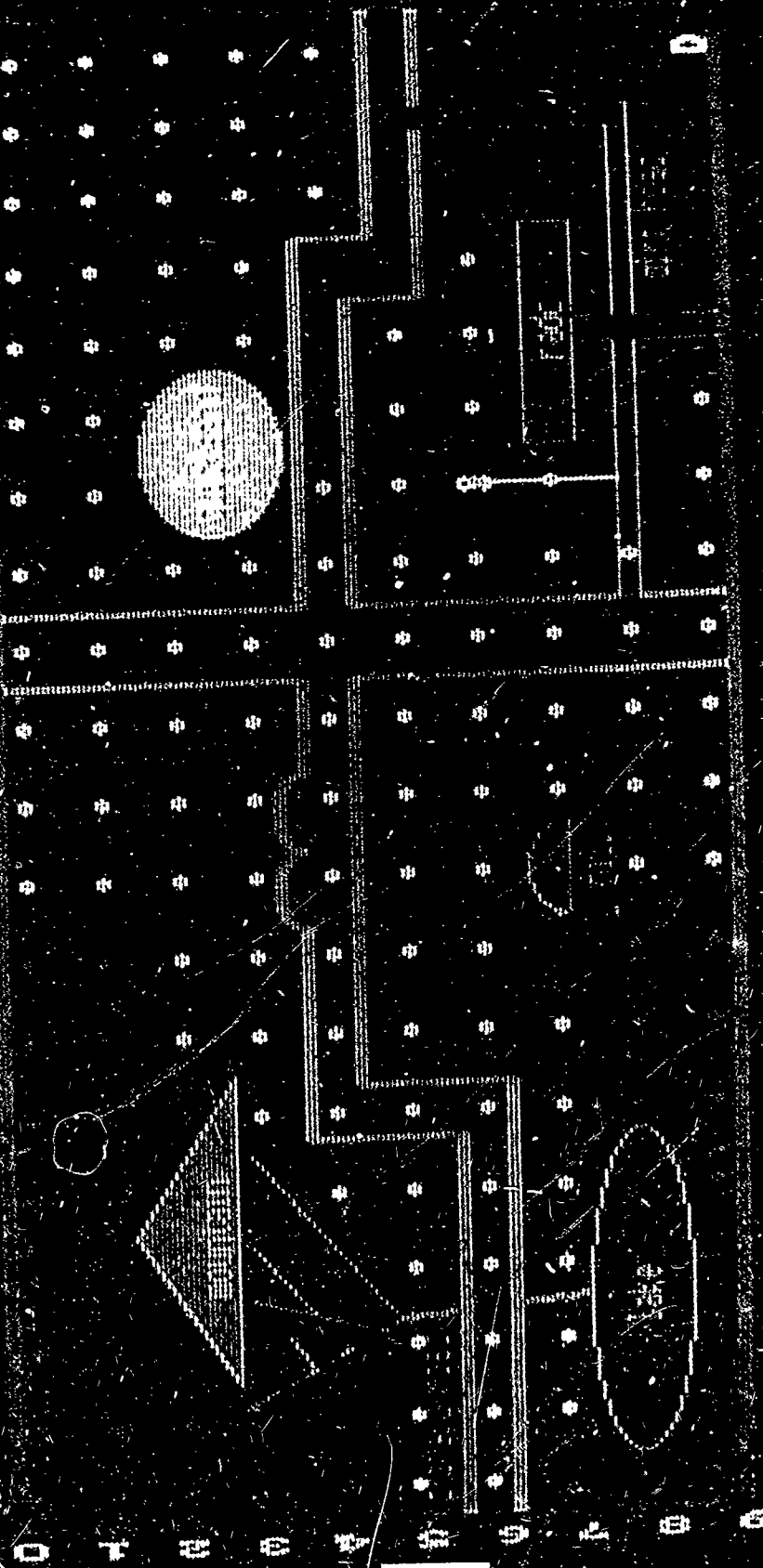


Figure 10. OPA 1.0 sample run N1 - C3.

Optimal Path Analyzer

A B C D E F G H I J K L M N O P Q R S T



Favorable area

Figure 11. OPA 1.0 sample run N6 - T9.

Optimal Path Analyzer

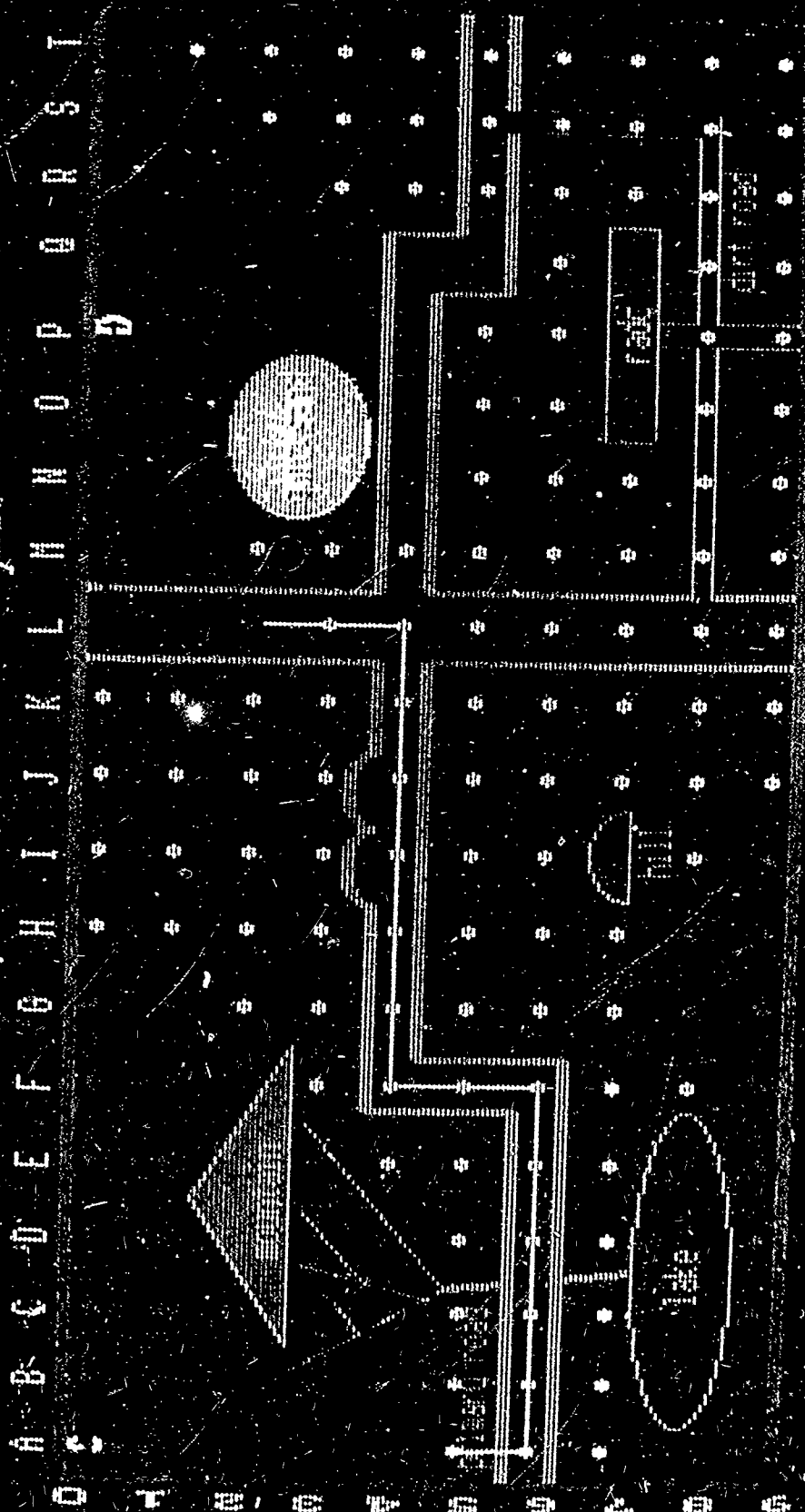


Figure 12. OPA 1.0 sample run A0 - P0.

Optimal Path Analyzer

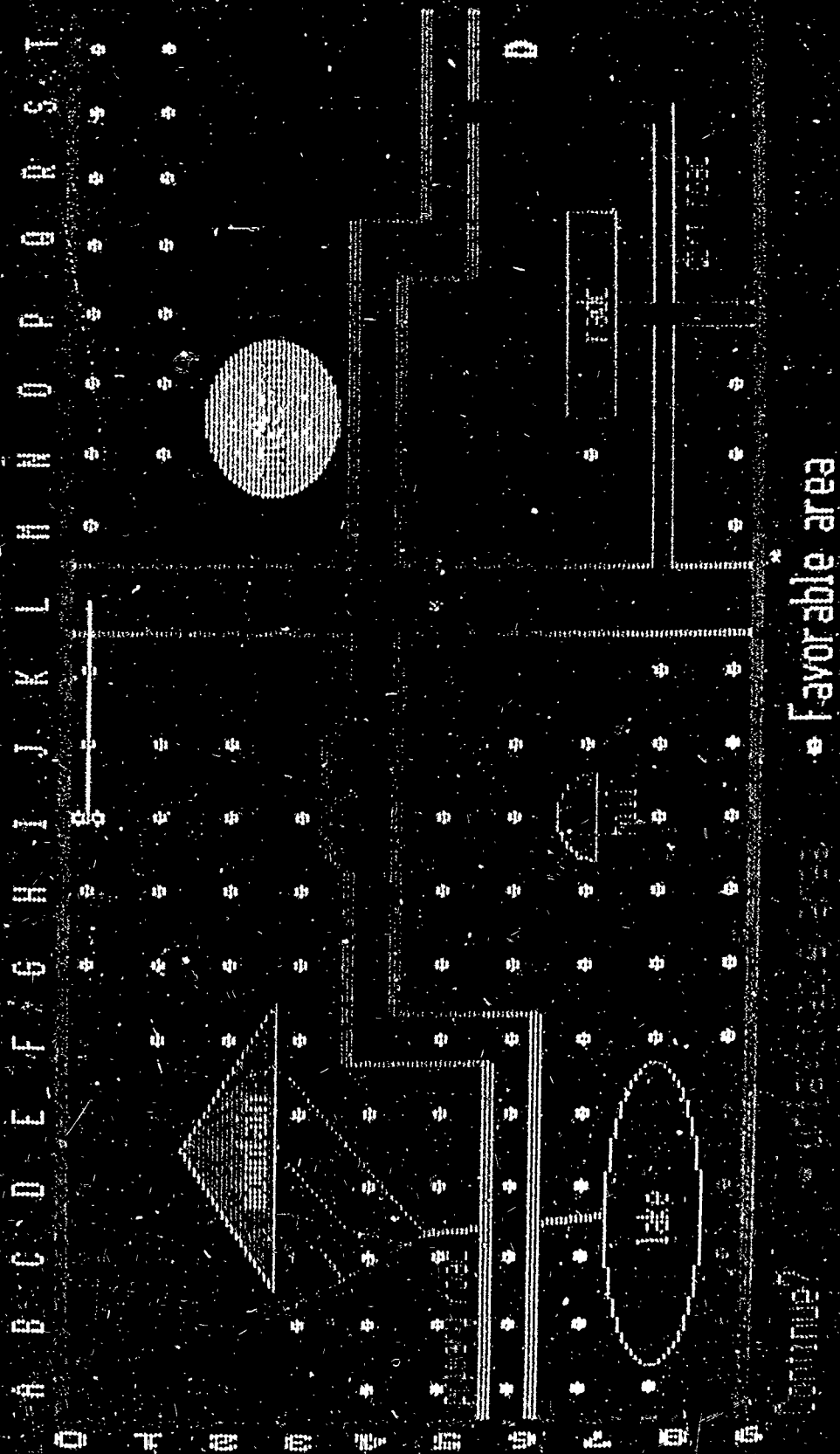
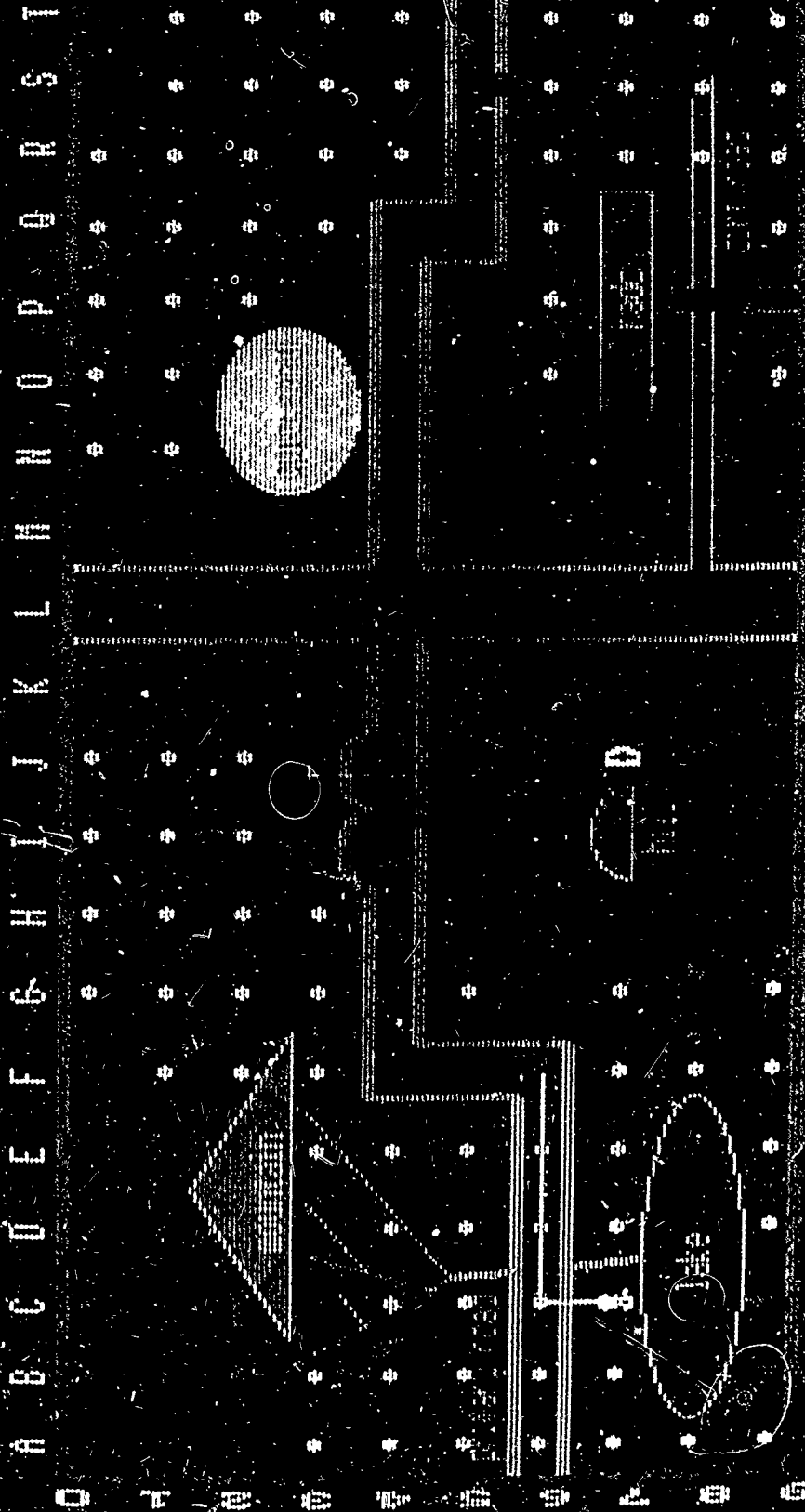


Figure 13. OPA 1.0 sample run IO - T6.

Optimal Path Analyzer

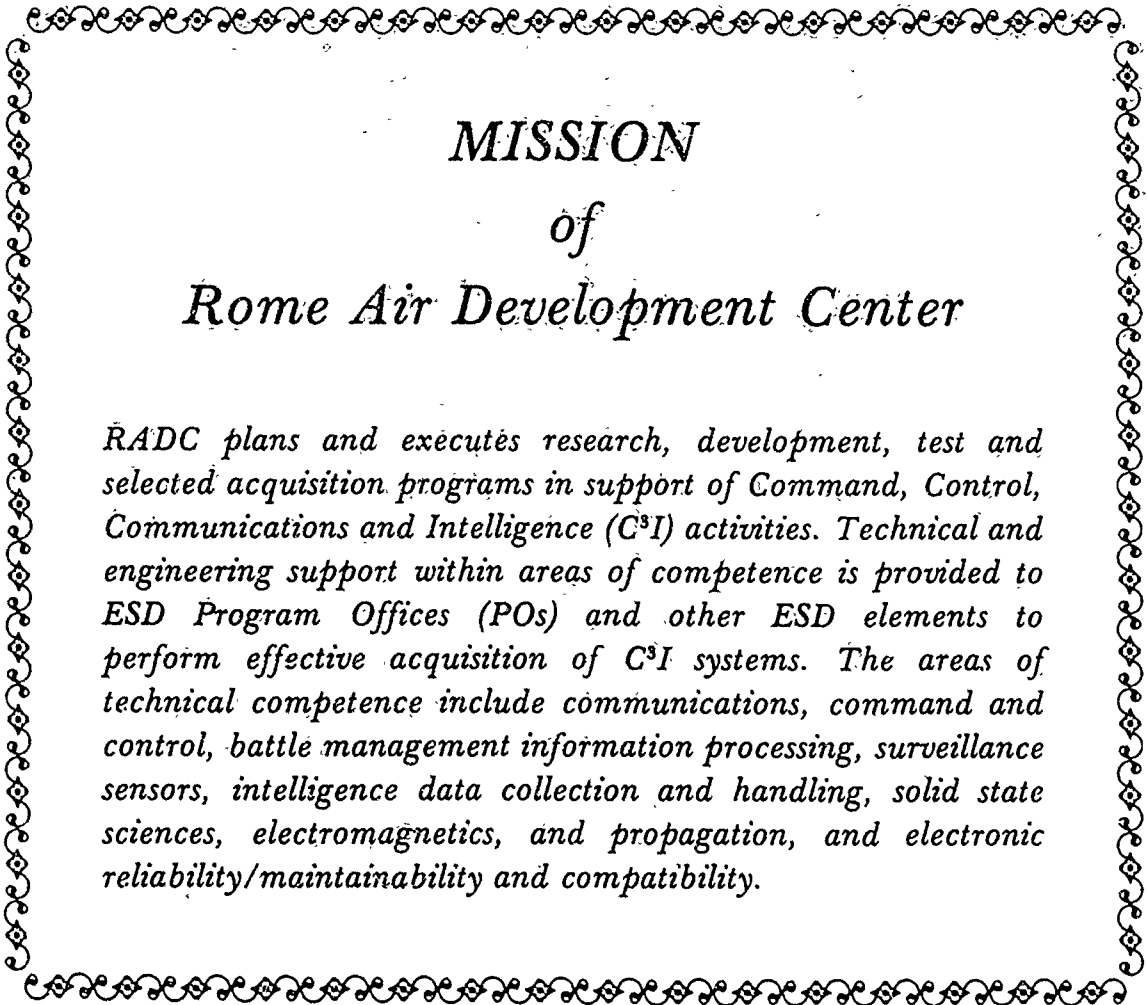


• Favorable area

Figure 14. OPA 1.0 sample run C7 - J7.

VII. References

- Bayle, Aime, "Learning in Neural Networks," PC AI, November/December, pp. 40-44.
- Bernstein, J., "Profiles: AI, Marvin Minsky," The New Yorker, pp. 50-126.
- Eliot, Lance B., "Neural Networks, Part 1: What are they and why is everybody so interested in them now?", IEEE Expert, Winter 1987, pp. 10-14.
- Freundlich, Naomi J., "Brain-Style Computers," Popular Science, February 1989, pp. 69-72, 110.
- Hebb, D. O., "The Organization of Behavior," John Wiley and Sons, New York, 1949.
- Hopfield, J. J., "Neural networks and physical systems with emergent collective computational abilities," Proceedings of the National Academy of Sciences, 79:2554-2558, 1982.
- Klimasauskas, Casimir C., "Neural Networks: A Short Course," PC AI, November/December, pp. 26-30.
- McCulloch, W. S., and W. Pitts, "A Logical Calculus of the Ideas Imminent in Nervous Activity," Bulletin of Mathematical Biophysics, 1943, Vol. 5, pg. 115-133.
- Minsky, Marvin, and Seymour Papert, "Perceptrons," The MIT Press, 1969, pg. 10.
- Rosenblatt, F., "Principles of Neurodynamics," Spartan Books, Washington, D.C., 1961.
- Sherald, Marge, "Neural Networks versus Expert Systems: Is There Room For Both?", PC AI, July/August 1989, pp. 10-15.
- Widrow, Bernard, Study Director, "DARPA Neural Network Study," October 1987 - February 1988, Lincoln Laboratory, MIT, Part II, pg. 5.



MISSION of Rome Air Development Center

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control, Communications and Intelligence (C³I) activities. Technical and engineering support within areas of competence is provided to ESD Program Offices (POs) and other ESD elements to perform effective acquisition of C³I systems. The areas of technical competence include communications, command and control, battle management information processing, surveillance sensors, intelligence data collection and handling, solid state sciences, electromagnetics, and propagation, and electronic reliability/maintainability and compatibility.